Computation-intensive Calculation Techniques

F. James, CERN

Suppose that we face the everyday problem of finding a given name in a telephone directory. There are, say, 10^6 names in the directory (in a single ordered list) and it takes one second to compare one name in the directory with the search item. Using the fact that the names are in alphabetical order, we could apply the binary search[1] technique, which would guarantee finding the name within about $\log_2(10^6) \sim 20$ compares, or 20 s. Indeed the strategy used by most human beings would be based largely on the binary search, modified by various considerations such as the person's knowledge of the relative frequency of occurrence of different letters. The human does not even consider doing a complete sequential scan through the whole book, since he knows that could take as long as 10^6 s — two weeks.

Suppose however that we have available a computer that performs one compare operation per microsecond. Now, even a sequential search which is 5×10^4 times slower than the binary search, would still take only one second, and certainly presents some clear advantages: Names no longer need be in order, searching can be done with incomplete information or based on other search keys besides the name, and multiple occurrences are easily detected.

The above very simple example illustrates some typical properties of the computation-intensive techniques made feasible by modern computers:

- Although requiring a lot of work by the computer, they are often conceptually simpler and easier to program than traditional methods designed for human performance.
- They are more easily adapted to modified situations (such as names not being in order) where the traditional method may depend critically on the accuracy of certain assumptions.
- They may allow a completely new way of looking at the problem and lead to radically new methods.

The most well-known computation-intensive methods are those known collectively as Monte Carlo (for a review, see [1]). Methods of this type have been known for centuries, but the most recent rediscovery which led to their widespread use in scientific calculations was probably due to Stan Ulam and John Von Neumann.

Monte Carlo was at first scorned by mathematicians as being devoid of any intellectual content. Then as its practical importance grew, a body of theory and specialized techniques was built up, which began to excite the interest of pure mathematicians who later were instrumental in advancing our understanding of the method. Generating "random" numbers on the computer, an essential component of any Monte Carlo method, was at first treated in an *ad hoc* manner, until it was discovered that many calculations were inaccurate because of the behaviour of the random number generator. Further investigation showed that it was not sufficient to make a generator complicated; on the contrary, the complications grafted onto bad random number generators invariably made them worse. The only way out was to try and understand the properties of the generating algorithms; this meant a return to simpler generators and raised some deep mathematical problems, especially in number theory, which are not entirely solved today. However some generators, including the most popular multiplicative congruential type, are now well enough understood to assure accurate Monte Carlo solution of many important problems.

Monte Carlo calculations are certainly computation-intensive, since they involve repeating identical calculations many times with different random values of certain variables, something one would not want to do by hand. And since the results converge to the desired answer only as fast as $1/\sqrt{N}$, where N is the size of the random sample, it is often necessary to let the programs run for a considerable time. Still for some problems, such as numerical integration of functions over spaces of high dimensionality, the Monte Carlo method actually converges faster than any known traditional method. Furthermore, the difficulties involved with the generation of pseudorandom[2] numbers, have led to the study of *quasirandom* sequences, which do not have to appear random in all respects, but must have the appropriate properties to obtain an accurate solution to the problem at hand. Whereas traditional numerical quadrature is based on such ideas as series expansions and orthogonal polynomials, and the usual Monte Carlo theory is based on the statistics of random variables, the theory of Quasi-Monte Carlo is based on the study of the distribution of large or infinite point sets in multidimensional hypercubes. The foundations for this theory, including the basic theorems about the discrepancy (deviation from uniformity) of point sets, were laid down by Hermann Weyl and others long before they could be put to practical use in computers. Although the practical side still presents a considerable challenge, and it is not yet known how to obtain asymptotic convergence rates as fast as the $(\log N)^d/N$ believed to be theoretically possible (where d is the dimensionality), quasirandom sequences are now used to obtain considerably improved convergence in Monte Carlo calculations. Here then is an example of a domain where the use of computation-intensive techniques, certainly very inefficient for one-dimensional problems, has led to the discovery (or rediscovery) of the most efficient techniques known for more difficult problems.

A field which has been revolutionized by the availability of powerful computers is that of statistical data analysis, so important in all experimental science. For many decades there has been an interesting parallel between the development of experimental science and that of statistical methodology, and the supercomputer era is no exception. Just as computers have allowed a giant step forward in the quality and quantity of experimental data, they have permitted the advent of completely new statistical techniques for analyzing the data, especially when it is lacking in quality or quantity.

In the pre-computer era, any statistical method had to be general enough to apply to a broad class of situations so that the calculation of the statistical significance could be done once and for

[1] Compare the name in the middle of the list with the target, thus identifying in which half of the list the target is to be found, and continue in the same way with the remaining half, thus eliminating half the remaining names at each step.

[2] Numbers generated by a computer algorithm, so they are not truly random, but should appear to be truly random to someone who does not know the algorithm.

all and published in a table. That is, it was implicit that all methods would be distribution-free. A good example is the χ^2 test for goodness of fit of a smooth curve to a data histogram. The significance level can be read from a pre-calculated table of χ^2, depending only on the number of histogram bins (the "degrees of freedom"), but not on the functional form fitted. Very nice, except that real data have a nasty habit of failing to conform to the requirements of the standard test: The bin contents may not be independent, or not Poisson distributed, or not large enough (the test is only *asymptotically* distribution-free). This is no obstacle to the physicist armed with a computer since he needs no table, and need not even worry about whether his test is distribution-free; he simply determines "empirically" the distribution of the test statistic under the actual experimental conditions, by generating random data according to the null hypothesis.

An even more radical departure from traditional statistics is represented by a class of methods known as *bootstrap*. The word comes from the expression "to pull yourself up by your own bootstraps", and refers to the use of the data sample itself to calculate the significance of the test. This technique is especially useful for comparing samples whose underlying distribution is unknown, which is nearly always the case with biological data, and often is true also in the more mathematical sciences.

A typical bootstrap situation would be to test whether the very highest energy cosmic rays observed in the northern hemisphere are significantly more energetic than those observed in the southern. Suppose we have set the energy cutoff so high that the sample consists of only 13 northern and 6 southern events. The straightforward approach would be to calculate the difference between the mean energies of the two samples (call this Δ_r) and try to determine whether this value is significantly different from zero. Traditionally one would apply the Student t-test, which is exact and optimal, but only when the underlying energy distributions are Gaussian. In the case at hand, the distributions are not known, but there is no reason for them to be Gaussian, so the traditional test would not be very meaningful.

Using the bootstrap, first calculate Δ_r as before, then forget which rays came from which hemisphere, pool the 19 events together and consider all the possible ways of dividing these 19 energy values into two samples of 13 and 6. For each of the 27 132 combinations, calculate Δ, the difference between mean energies, and let $g(\Delta)$ be the number of combinations with energy difference Δ. Under the null hypothesis (no difference between hemispheres), each combination is equally probable. This means that the sum of $g(\Delta)$ from Δ_r to infinity, divided by 27 132, is directly the confidence level in the usual statistical sense: the probability, under the null hypothesis, of observing a difference Δ greater than Δ_r. The method is immediately understood intuitively, takes account of all the observed quantities, and does not require any additional unjustified assumptions. It is also possible to use a distance measure other than the difference of mean energies, if such a measure is more sensitive to the anisotropy predicted by a theory. And if the samples are so large that even a computer cannot enumerate all possible combinations, the method can be modified to sample randomly the different combinations until a sufficiently accurate approximation for $g(\Delta)$ is obtained.

Applications of computation-intensive techniques in theoretical physics are too numerous to discuss here in any detail, but some mention must be made of what is perhaps the most spectacular such application, *QCD on the lattice* [2]. Pioneered by Ken Wilson, this allows in principle calculation of the most fundamental properties of matter, such as the masses of the elementary particles, from first principles of Quantum Chromo Dynamics. The lattice is a discretization of four-dimensional space-time, and must have enough points to approximate the continuum. One then uses a Monte Carlo method to find states of statistical equilibrium with respect to fluctuations of the complex quantum field. Even the least ambitious QCD lattice calculations strain the resources (both memory space and computing speed) of the world's most powerful computers, a situation recognized very early on, and leading many of the researchers in the field (including Wilson himself) to spend a large part of their time designing computer hardware specially adapted to solving this problem.

A cynic might remark that all this is just another manifestation of Parkinson's law: that as computers get bigger, physicists think up new ways to keep them busy. I prefer to look at these applications as exciting new approaches to the solution of physical problems previously considered intractable.

REFERENCES

[1] James F., 'Monte Carlo Theory and Practice', *Rep. Prog. Phys.* **43** (1980) 1145-1189.
[2] Creutz M., Rebbi C. and Jacobs L., 'Monte Carlo computations in lattice gauge theories', *Phys. Rep.* **95** (1983) 4, 201.

Experimental Physics Control Systems

Inter-Divisional Group

From 28 September to 2 October 1987, Villars-sur-Ollon (Switzerland) hosted the international conference on control systems for experimental physics organized by the EPS Interdivisional Group on Experimental Physics Control Systems (EPCS). Over a full week, scientists from all over the world discussed the problematics of controlling processes as complex as particle accelerators, nuclear fusion devices, large telescopes and high energy physics detectors.

Close to 80 contributed papers were accepted; most of which were displayed in poster sessions. Invited papers gave overviews of major control systems, supplemented by contributed papers, workshops, tutorials and exhibits. Panel workshops covered a substantial part of the programme. Authors with different opinions were invited to the podium to defend their points of view, often leading to lively discussions. There were workshops on Architecture, Interfacing and Intelligent Process Equipment, Applications Software, Operational Aspects, Automation, Software Engineering. A final brain storming on Contemporary and Future Main Issues, terminated this series of highly animated debates.

The trend towards more complex machines leads to more complex controls. On the other hand, data processing technology progresses rapidly and the cost of hardware decreases steeply. No surprise that the current trend in controls is towards fully distributed systems with more and more local intelligence at the device level. In broad lines, the control architectures are based on two level networks: a backbone, token ring or